Midterm 2 (13.12.2018) (08:30-09:45)

COMP303 – Computer Architecture

Q1(10)	Q2(15)	Q3(15)	Q4(20)	Q5(15)	Q6(10)	Q7(15)	Q8(15)	Q9(00)	Q10(00)	Total

Q1. Classify and briefly describe I/O methods that can be used in a computing system. (10)

A1.

I/O can be controlled in four general ways.

Programmed I/O:

Reserves a register for each I/O device.

Each register is continually polled to detect data arrival.

Interrupt-Driven I/O:

Allows the CPU to do other things until I/O is requested.

Direct Memory Access (DMA):

Offloads I/O processing to a special-purpose chip that takes care of the details.

Channel I/O:

Uses dedicated I/O processors.

==========

Q02. Suppose a system has a virtual memory size of 64K and the system uses word addressing. If the page size is 1024 Word,

a. How many bits this virtual memory address requires? (07)

b. Determine the number of virtual pages. (08)

A02.

virtual address space = 64K = $2^6 \times 2^{10} = 2^{16}$ Word

page size = 1024 Word = 2^{10} Word

a. Because the virtual memory size is 2^{16} Word, the virtual memory address requires **16 bits**

b. Number of virtual pages = $2^{16} / 2^{10} = 2^{16-10} = 2^6 =$ **64**

==========

Q03. Assume that you have a computer with a maximum main memory size of 2 GByte (motherboard does not allow to upgrade the main memory). The program you want to run on this computer may require a larger main memory, resulting in a virtual memory that uses part of the hard drive. However, you realized that the magnetic hard drive in this computer is too slow for your application.

What can you do to increase the performance of this computer? Explain how and why your solution will increase the performance. (15)

A03.

I would replace magnetic hard drive with a Solid State Drive (SSD). SSDs have the following advantages over HDDs:

High-performance input/output operations per second (IOPS), less susceptible to physical shock and vibration longer lifespan, lower power consumption, lower access times and latency rates.

==========

Q04. A 16 bit microprocessor system has 10 address lines to address relevant memory locations.

a. Assuming that the data size is 1 byte, what is the address of the last memory location? (05)
b. Assuming memory chips organized as 512x8 bits are used, how many memory chips we need to design the required memory). (05)
c. Design the required memory system using the memory chips organized as 512x8 bits. (10)

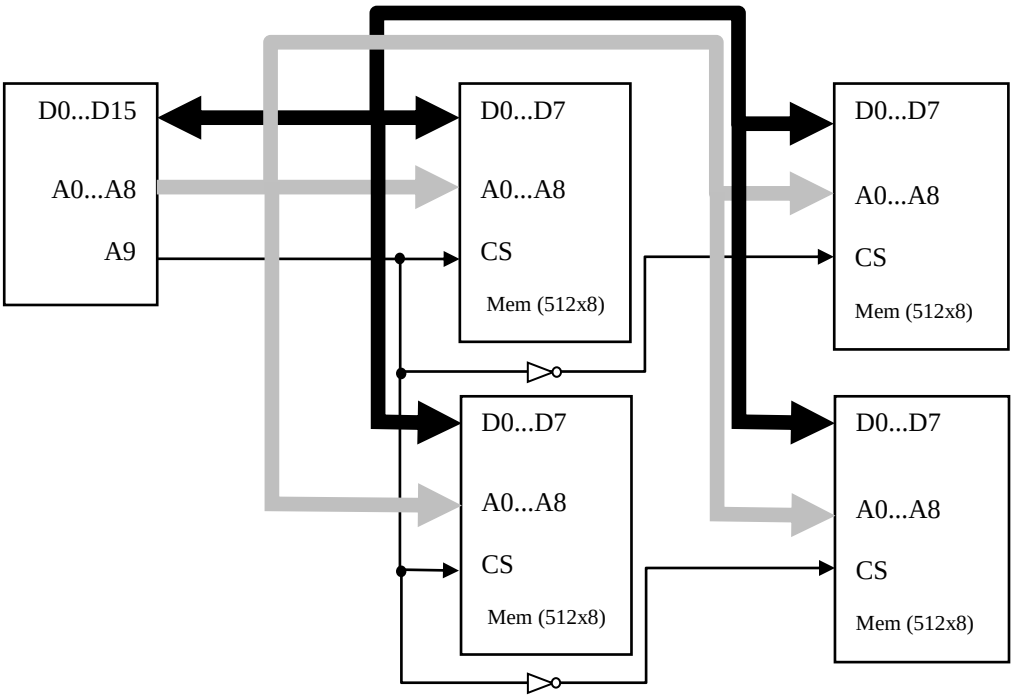
A04.

a. Assuming that the data size is 1 byte, what is the address of the last memory location? (05)
The last memory location **= $2^{\#\text{of bits in address line}}$ – 1 = 2^{10} – 1 = 1023**
b. Assuming memory chips organized as 512x8 bits are used, how many memory chips we need to design the required memory). (05)

Data bus size **16 bits**, memory size is **1024**. System memory organization is **1024x16**. If we restate this memory organization in terms of the available memory chip (512x8), we have {(2x512)x(2x8)} = **4**x(512x8).

So, We need **4** (512x8) memory chips to have a memory organized as 1024x16

c. Draw a block diagram of the computing system by designing the required memory system using the memory chips organized as 512x8 bits. (10)



==========

Q05. Consider a processor which uses branch delay technique for optimization of pipelining. Each intruction has 2 or 3 stages depending on the instruction type (I: Instruction fetch, E: Execute D: Memory operation)

100 LOAD	**R1, (500)**
101 JUMP	**105**
102 ADD	**R1, 1**
103 AND	**R1, (501)**

a. Which instruction will be executed first after the processor encounters the instruction in line 101. (07)

b. Complete the following pipelinening diagram for the given program. (08)

A05. The solution is as follows:

a. First, instruction in line 100, which adds contents of memory location 500 to R1 will be executed. In line 101 there is a jump (unconditional branch) instruction. Because the processor uses a branch delay technique, instruction in line 102 will be executed before processor jumps to line 105.

b.

		1	2	3	4	5	6
100 LOAD	**R1, (500)**	I	E	D			
101 JUMP	**105**		I	E			
102 ADD	**R1, 1**			I	E		
103 AND	**R1, (501)**				I	E	D

==========

Q06. Why a **cash** and **virtual memory** are used in a computing system? Briefly explain. (10)

A06.

Cache memory enhances performance by providing faster memory access speed. The purpose of cache memory to speed up accesses by storing recently used data closer to the CPU. It is much smaller than main memory. Its access time is a fraction of that of main memory.

Virtual memory enhances performance by providing greater memory capacity, without the expense of adding main memory. Instead, a portion of a disk drive serves as an extension of main memory.

==========

Q07. If one line of an assembler program (which loads contents of memory location 7400 to the accumulator) for a hypothetical processor and corresponding machine code is given as:

address	assembler	machine code	
3000	lda 7400	1A7400	(the values are in hexadecimal)

a. How many bits are needed for the program counter?

b. How many bits are needed for the instruction register?

c. How many instructions this processor might have?

d. For the memory location 7400, find the Tag, Line, and Word values in hexadecimal format for a direct-mapped cache, when **tag-id = 5 bits**, **line-id = 7 bits**, **word-id = 4 bits**.

A07.

a. 16 Bits

b. 24 Bits

c. 256

d. 7400 ➔ 0 1 1 1 0 1 0 0 0 0 0 0 0 0 0 0

tag-id line-id word-id

tag-id = 0 0 0 0 **1 1 1 0** = 0E

line-id = 0 1 0 0 0 0 0 0 = 40

Word-id= 0 0 0 0 = 0

==========

Q08. Following table shows a small part of memory in a computer system. What would be the contents of accumulator after executing each instruction in the following assembler code. (15)

100	**LOAD** *IMMEDIATE*	**A1**	Acc = ?	
101	**XOR** *IMMEDIATE*	**FF**	Acc = ?	
102	**ADD** *INDIRECT*	**A4**	Acc = ?	
103	**OR** *INDIRECT*	**A1**	Acc = ?	
104	**SUB** *DIRECT*	**A0**	Acc = ?	

Mem. Adress	Data
A0	11
A1	A3
A2	22
A3	3A
A4	A2

A08.

100	**LOAD** *IMMEDIATE*	**A1**	Acc = A1	1 0 1 0 0 0 0 1
101	**XOR** *IMMEDIATE*	**FF**	Acc = 5E	0 1 0 1 1 1 1 0
102	**ADD** *INDIRECT*	**A4**	Acc = 80	1 0 0 0 0 0 0 0
103	**OR** *INDIRECT*	**A1**	Acc = BA	1 0 1 1 1 0 1 0
104	**SUB** *DIRECT*	**A0**	Acc = A9	1 0 1 0 1 0 0 1